Exercise 6
Heaps

Exercise Objectives:
1. To write a program to Heap tree (max) and perform insertion and deletion operations on it implemented in array or linked-list.

Additional Requirement:
1. Provide the flowchart of the entire program.

Heap Tree

A max-heap is a complete or nearly complete binary tree in which the value in each internal node is greater than or equal to the values in the children of that node.

Hands-on Exercises:

[12_Heaps]

Create a heap tree using linked representation with having the structure

Linked List Implementation

```
typedef struct node
{
    int data;
    struct node *next;
}NODE;
```

```
typedef struct tree
{
    int count;
    struct node *root;
    struct node *lastleaf;
}TREE;
```

The program would be able to perform the following operations:

[1] Insert Node
[2] Delete Node
[3] Exit

** Display tree every time insertion and deletion is performed*

OPERATION: Insert Node

To add an element to a heap it must perform an up-heap operation (also known as bubble-up, percolate-up, sift-up, trickle up, heapify-up, or cascade-up), by following this algorithm:

```
Algorithm insertNode(tree, data)
1. Add the element to the bottom level (last leaf) of the heap.
2. Compare the added element with its parent; if they are in the correct order, stop.
3. If not, swap the element with its parent and return to the previous step.
end insertNode
```

OPERATION: Delete Node

The procedure for deleting the root from the heap (effectively extracting the maximum element in a max-heap or the minimum element in a min-heap) and restoring the properties is called down-heap (also known as bubble-down, percolate-down, sift-down, trickle down, heapify-down, cascade-down and extract-min/max).

```
Algorithm deleteNode(tree)
1. Replace the root of the heap with the last element on the last level.
2. Compare the new root with its children; if they are in the correct order, stop.
3. If not, swap the element with one of its children and return to the previous step. (Swap
   with its smaller child in a min-heap and its larger child in a max-heap.)
end deleteNode
```